Uncertainty estimation via ROCKET

Alexey Voskoboinikov[1]

Abstract

Uncertainty estimation via ROCKET.

1. Introduction

Modern machine learning methods including deep learning have achieved great success in predictive accuracy for supervised learning tasks, but may still fall short in giving useful estimates of their predictive uncertainty (Ovadia et al., 2019). In this project, I'll reproduce ROCKET time series classification method and asses different uncertainty estimates for it.

2. Related work

2.1. Uncertainty estimate

Classification models usually predict some score, that could be treated as confidence. The simplest way to quantify uncertainty is to use probability of the complement outcome:

$$f_{ue}(x) = 1 - \max_k p_\theta(y = k \mid x)$$

Unfortunately, this approach does not reflect the true uncertainly, as models tend to be overconfident (Guo et al., 2017).

2.2. Quality of the Uncertainty estimate

To evaluate quality of predictive uncertainty, commonly metrics are Negative loglikelihood, Brier Score, Expected Calibration Error (Ovadia et al., 2019). their descriptions are given below.

2.2.1. Negative loglikelihood

is a proper scoring rule and a popular metric for evaluating predictive uncertainty (Lakshminarayanan et al., 2017). It's main drawback is that it can over-emphasize tail probabilities.

$$\textbf{NLL} = -\frac{1}{N}\sum_{n=1}^{N} \log p_\theta(y_n \mid x_n)$$

2.2.2. Brier Score

is also a proper scoring rule for measuring the accuracy of predicted probabilities. It is computed as the squared error of a predicted probability. But, it is insensitive to predicted probabilities associated with in/frequent events.

$$\textbf{BS}(y^*, x^*) = \frac{1}{K}\sum_{k=1}^{K} \left(\mathbf{I}\left[k = y^*\right] - p_\theta\left(y = k \mid x^*\right)\right)$$

.

2.2.3. Expected Calibration Error

measures the correspondence between predicted probabilities and empirical accuracy. It is computed as the average gap between within bucket accuracy and within bucket predicted probability for S buckets $B_s = \{n \in 1 \ldots N : p(y_n|x_n, \theta) \in (\rho_s, \rho_{s+1}]\}$.

$$\textbf{ECE} = \sum_{s=1}^{S} \frac{|B_s|}{N} |\texttt{acc}(B_s) - \texttt{conf}(B_s)|.$$

$$\texttt{acc}(B_s) = |B_s|^{-1} \sum_{n \in B_s} [y_n = \hat{y}_n].$$

$$\texttt{conf}(B_s) = |B_s|^{-1} \sum_{n \in B_s} p(\hat{y}_n|x_n, \theta).$$

$$\hat{y}_n = \arg\max_y p_\theta(y|x_n).$$

When bins $\{\rho_s : s \in 1 \ldots S\}$ are quantiles of the held-out predicted probabilities, $|B_s| \approx |B_k|$ and the estimation error is approximately constant. Due to binning, ECE does not monotonically increase as predictions approach ground truth. If $|B_s| \neq |B_k|$, the estimation error varies across bins.

3. Dataset Description

To test ROCKET algorithm and quality if uncertainty estimate, I used UCR Time Series Classification Archive (Dau et al., 2018). It is the set of 85 time series datasets from different domains. Each datasets has time series of fixed length, with predetermined train/test split.

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

4. ML Methods and algorithms

4.1. Random Convolutional Kernels

For each dataset of time series with fixed length l, separate set of random kernels is created. Each kernel is created by sampling or evaluating it's parameters as follows:

- **kernel length**: $k \sim \mathcal{U}\{7, 9, 11\}$.
- **weights**: $w' \sim \mathcal{N}(w \mid 0, I), \quad w = w' - \mathbf{mean}(w')$.
- **bias**: $b \sim \mathcal{U}(-1, 1)$.
- **dilation**: $d = \lfloor 2^x \rfloor, \quad s \sim \mathcal{U}(0, A), \quad A = \log \frac{l-1}{k-1}$.
- **padding**: $p = \frac{d(k-1)}{2}$, used with probobility 0.5.

The default number of kernels N is 10000.

4.2. Uncertainty estimates

For each of the ways listed below, I evaluated NNL, DS and ECE as commonly used metrics of uncertainty estimate quality.

4.2.1. Single trial uncertainty estimate

As a baseline solution, I used single logistic regression, trained on the set of random features as multinational distribution by minimizing cross-entropy loss function. Optimization was performed in PyTorch with Adam optimizer with learning rate 10^{-4} and weight decay equal 1, which is equivalent to L_2 regularization. This is convex optimisation problem, so optimization continued until either loss decreasing per step was less then 10^{-4} over the interval, or 5000 epochs.

4.2.2. Ensemble uncertainty estimate

Creating ensemble of the model is in some close to Monte Carlo estimation of predictive posterior, as predictive distribution is averaged over the parameters of the models in the ensemble. Here? to build ensemble of the models, I created $M = 10$ sets random features, trained logistic regression on each set individually, and averaged predictions:

$$p(y^* \mid x^*, D) = \mathbf{E}_{\theta \sim p(\theta \mid D)} \left[p(y^*, \mid x^*, \theta) \right]$$

$$\approx \frac{1}{M} \sum_{m=1}^{M} p(y^*, \mid x^*, \theta), \quad \theta \sim p(\theta \mid D).$$

4.2.3. Dropout uncertainty estimate

Another way to get better uncertainty estimate is to use dropout on random features during evaluation, evaluate model multiple times and average predictions (Gal & Ghahramani, 2016). This is close to the creating an ensemble of the model, as each model uses only subset of the features, and this way it is a small tweak in the model parameters.

4.2.4. FFT-based uncertainty estimates

Before this stage, I only used MC estimation of predictive distribution by estimate or dropout, which result in integration over the parameters of the model. In the next approach, I introduced uncertainty into the data during evaluation and averaged over this uncertainty, while treating parameters of the model as a point estimate. To do this, I needed the way to resample from distribution of time series. I assumed that the time series came from Gaussian process with fixed amplitude and random phase of the spectrum. The sampling procedure was as follows: apply Fourier transform to time series, changed phase at the randomly chosen frequency, and returned to the time domain with inverse Fourier transform.

Table 1. Performance of uncertainty estimates.

Uncertainty estimate	NLL	BS	ECE
STE	2.5	2.25	2.44
EE	2.37	2.41	2.51
MCDE	2.38	2.46	2.33
FFTE	2.76	2.87	2.72

5. Experiments

5.1. Reproduced ROCKET model

First, to asses quality of the reproduced model, I compared accuracy of the model on the time series presented by the authors (Dempster et al., 2019) of original ROCKET paper and the reproduced model. In comparison of Ridge Classifiers, I achieved Pearson correlation coefficient $r = 0.991$ (Figure 4). Ridge Classifier is not a probabilistic model, so I compared accuracy of Ridge Classifiers from the paper and reproduced logistic regression and got $r = 0.892$ (Figure 5), and accuracy of logistic regression was considerably worse.

5.2. Uncertainty estimates

I compared 4 types of uncertainty estimates (STE - Single trial uncertainty estimate (baseline), EE - Ensemble uncertainty estimate, MCDE - Monter Carlo dropout uncertainty estimate, FFTE - FFT-based uncertainty estimates) with 3 matrics (see Table 1). FFTE was the worst estimate on the all metrics (even compared to the baseline), EE and MCDE outperformed STE on the NLL metric, and MCDE outperformed on the ECE metric. Overall, MCDE showed better results then other estimates (Figure 1, 2, 3).

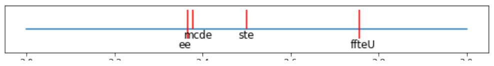

Figure 1. Negative loglikelihood.

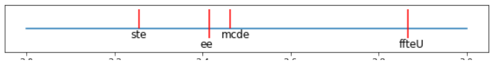

Figure 2. Brier Score.

6. Discussion

In the original paper ROCKET uses ridge classifier, which is not a probabilistic model. As pointed out in the paper, it's faster to train then logistic regression and it turned out to have higher accuracy, but makes uncertainty estimation challenging. Substitution of logistic regression instead of ridge classifier reduces accuracy, but makes common ways to asses uncertainty possible. Monte Carlo Dropout turned out to produce better results then other methods of uncertainty estimation. Unfortunately, my way to resample from the time series did not produced good results, and it possible that there are much better ways to do that.

7. Conclusion

References

Dau, H. A., Bagnall, A. J., Kamgar, K., Yeh, C. M., Zhu, Y., Gharghabi, S., Ratanamahatana, C. A., and Keogh, E. J. The UCR time series archive. *CoRR*, abs/1810.07758, 2018. URL http://arxiv.org/abs/1810.07758.

Dempster, A., Petitjean, F., and Webb, G. I. ROCKET: exceptionally fast and accurate time series classification using random convolutional kernels. *CoRR*, abs/1910.13051, 2019. URL http://arxiv.org/abs/1910.13051.

Gal, Y. and Ghahramani, Z. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pp. 1050–1059. PMLR, 2016.

Guo, C., Pleiss, G., Sun, Y., and Weinberger, K. Q. On calibration of modern neural networks. *CoRR*, abs/1706.04599, 2017. URL http://arxiv.org/abs/1706.04599.

Lakshminarayanan, B., Pritzel, A., and Blundell, C. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in neural information processing systems*, 30, 2017.

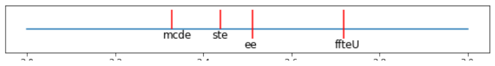

Figure 3. Expected Calibration Error.

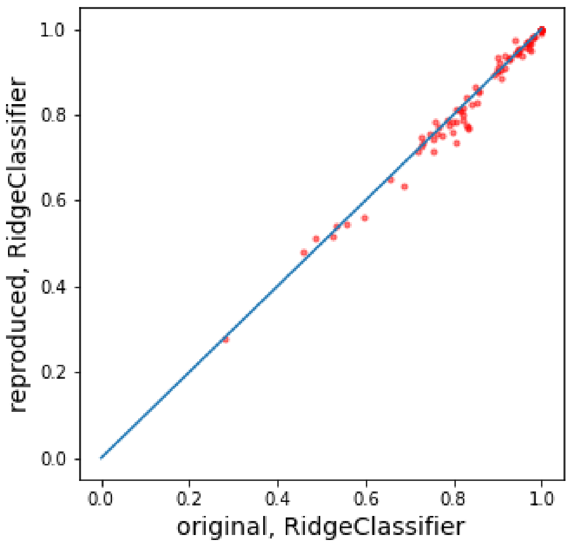

Figure 4. Accuracy or Ridge Classifier model in the original paper and it's reproduction. In ideal scenario red dots must lay on the blue line.

Ovadia, Y., Fertig, E., Ren, J., Nado, Z., Sculley, D., Nowozin, S., Dillon, J., Lakshminarayanan, B., and Snoek, J. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. *Advances in neural information processing systems*, 32, 2019.

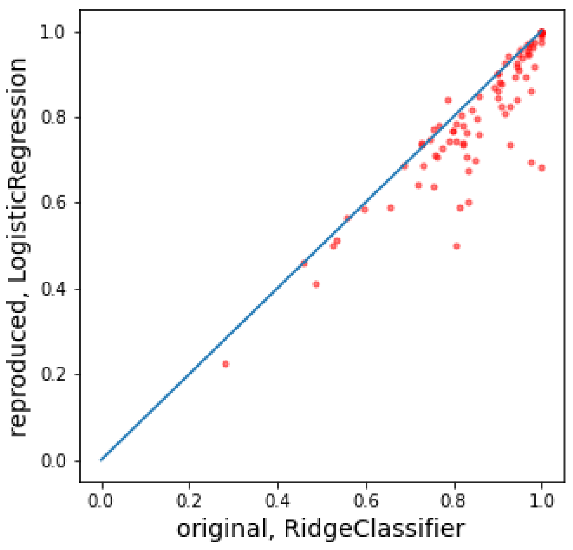

Figure 5. Accuracy or Ridge Classifier model in the original paper and reproduction of Linear Regression.